

Ingeniería de Software



Universidad Tecnológica Nacional
Facultad Regional Córdoba
Cátedra de Ingeniería de Software
Docentes: Judith Meles – Laura Covaro

Definición de Software

Software, en general, es un set de programas y la documentación que acompaña.

- Existen tres tipos básicos de software. Estos son:
 - System software
 - Utilitarios
 - Software de Aplicación

Margaret Hamilton, lead software engineer, Project Apollo. Mostrando su código fuente..

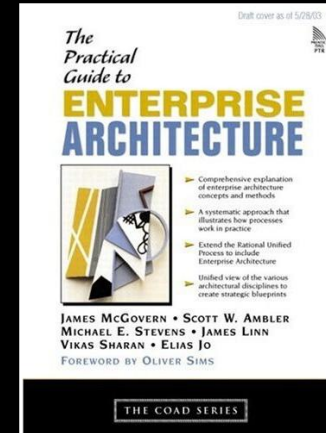
<https://medium.com/@3fingeredfox/margaret-hamilton-lead-software-engineer-project-apollo-158754170da8>



5 Razones para no comparar software y manufactura

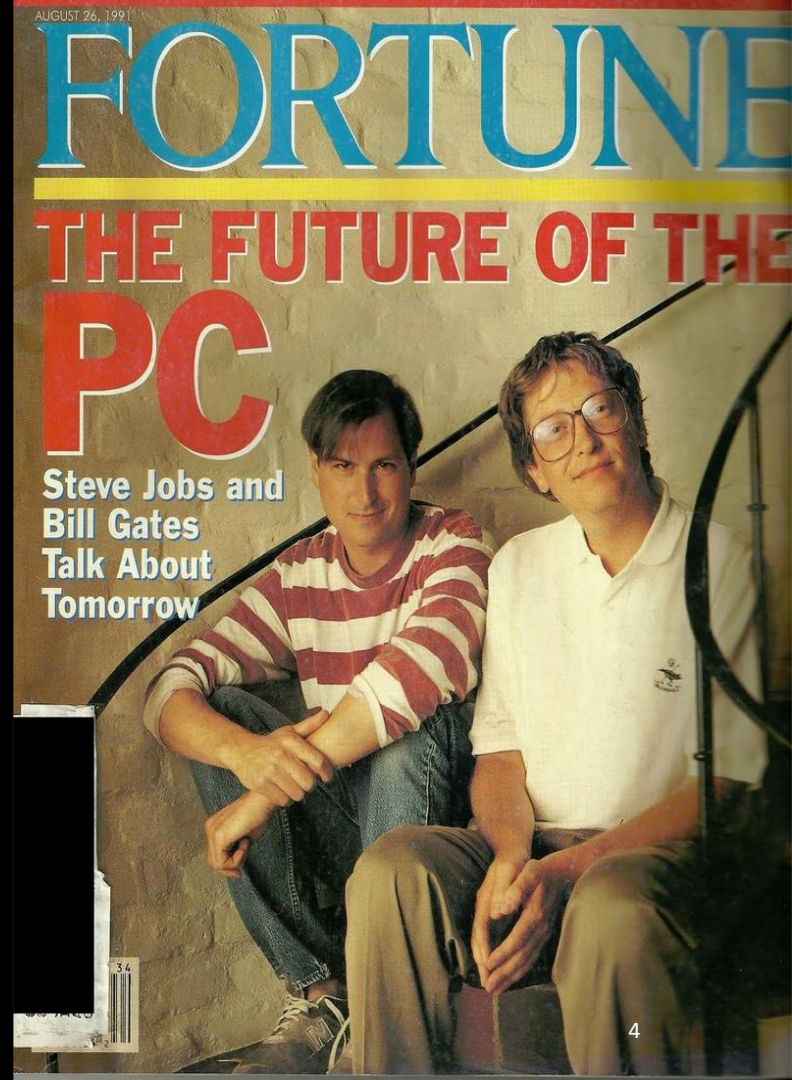
- ✓ El software es menos predecible
- ✓ No hay producción en masa, casi ningún producto de software es igual a otro.
- ✓ No todas las fallas son errores
- ✓ El software **no se gasta**
- ✓ El software no esta gobernado por las leyes de la física 😊

"The creation of software is an intellectual human endeavor. Creating good software relies on the personalities and the intellects of the members of the teams that create it. When applied to a different team of developers a process that delivers great software for one team of developers may fail to deliver anything at all for another team."
-- *The Practical Guide to S/W Arch.*



Un poco de historia

- 1968 • Nace el termino – conferencia de la NATO
- 1975 • The Mythical Man-Month – Frederick Brooks
- 1978 • Tom DeMarco introduce Structured Analysis
- 80' • Primeros grandes errores de software conocidos
- 1987 • No silver bullet (Brooks). Características esenciales del software
- 1989 • Managing the Software Process – Watts Humphrey
- 1990 • Internet / Object Oriented
- 1991 • CMM 1.0
- 1993 • CMM 1.1
- 2000 • CMMI 1.0
- 2001 • Agile Manifiesto
- 2003 • Lean Software Development



GLOBAL TOP 5 BRANDS | TECH ASCENDANCY

RANK

1

2

3

4

5

2006



2012



2017



Las 10 empresas más grandes del mundo en 2019

- 1 **Apple** (881.980 millones de dólares)
- 2 **Microsoft** (803.090 millones de dólares)
- 3 **Amazon** (739.460 millones de dólares)
- 4 **Alphabet** (711.940 millones de dólares)
- 5 **Berkshire Hathaway** (536.530 millones de dólares)
- 6 **Johnson & Johnson** (396.210 millones de dólares)
- 7 **Alibaba** (387.610 millones de dólares)
- 8 **Facebook** (378.050 millones de dólares)
- 9 **JPMorgan Chase & Co** (368.560 millones de dólares)
- 10 **Tencent Holdings** (353.670 millones de dólares)

Algunos problemitas con el desarrollo de software



La versión final del producto no satisface las necesidades del cliente.



No es fácil extenderlo y/o adaptarlo. Agregar más funcional en otra versión es casi una misión imposible



Mala documentación



Mala calidad.



Más tiempos y costos que los presupuestados

Errores informáticos más costosos de la historia

<https://www.javiergarzas.com/2013/05/top-7-de-errores-informaticos.html>

1 – La destrucción del Mariner I (1962). 18,5 millones de dólares.

El del Mariner I, una sonda espacial que se dirigía a Venus, se desvió de la trayectoria de vuelo prevista poco después del lanzamiento. Desde control se destruyó la sonda a los 293 segundos del despegue. La causa fue una fórmula manuscrita que se programó incorrectamente.

2 – La catástrofe del Hartford Coliseum (1978). 70 millones de dólares.

Apenas unas horas después de que miles de aficionados abandonaron el Hartford Coliseum, el techo se derrumbó por el peso de la nieve. La causa: cálculo incorrecto introducido en el software CAD utilizado para diseñar el coliseo.

3 – El gusano de Morris (1988). 100 millones de dólares.

El estudiante de posgrado Robert Tappan Morris fue condenado por el primer ataque con “gusanos” a gran escala en Internet. Los costos de limpiar el desastre se cifran en 100 millones de dólares. Morris, es hoy profesor en MIT.

4 – Error de cálculo de Intel (1994). 475 millones de dólares.

Un profesor de matemáticas descubrió y difundió que había un fallo en el procesador Pentium de Intel. La sustitución de chips costó a Intel 475 millones.

5 – Explosión del cohete Ariane (1996). 500 millones de dólares.

En el 1996, el cohete Ariane 5 de la Agencia Espacial Europea estalló. El Ariane explotó porque un número real de 64 bits (coma flotante) relacionado con la velocidad se convirtió en un entero de 16 bits. [Te dejo un post que explica lo sucedido.](#)

6 – Mars Climate Orbiter (1999). 655 millones de dólares.

En 1999 los ingenieros de la NASA perdieron el contacto con la Mars Climate Orbiter en su intento que orbitase en Marte. La causa, un programa calculaba la distancia en unidades inglesas (pulgadas, pies y libras), mientras que otro utilizó unidades métricas.

7 – El error en los frenos de los Toyota (2010). 3 billones de dólares.

Toyota retiró más de 400.000 de sus vehículos híbridos en 2010, por un problema software, que provocaba un retraso en el sistema anti-bloqueo de frenos. Se estima entre sustituciones y demandas el error le costó a Toyota 3 billones de dólares.

8 – Las migraciones por el año 2000. 296,7 billones de dólares.

Se esperaba que el bug Y2K paralizase al mundo a la medianoche del 1 de enero 2000, ya que mucho software no había sido previsto para trabajar con el año 2000. El mundo no se acabó, pero se estima que se gastaron 296,7 billones de dólares para mitigar los daños.

ANNOUNCEMENT The Next Web is coming back to New York! Learn more here.



Chrysler is recalling 1.4 million vehicles after a Jeep was remotely hacked

by NADER LOPEZ 25 Twitter — 3d ago in INSIDER



All Time Favorites

The complete guide to backing up your computer properly

How to use personal drones legally: A beginner's guide

11 Android apps to make notifications more interesting

How to raise prices without ruining your business

How mobile is bridging brick and mortar's data gap



http://t.me/nextweb

0 Comments

INFOSAE

Un error de software de algunos Airbus A350 requiere apagar los aviones cada 149 horas o actualizar el sistema

28 de julio de 2019

Compartir en Facebook

Compartir en Twitter



Hay una falla de software en los Airbus A350 que ya fue reparada en la actualización de software (Shutterstock)

MÁS LEÍDAS EN infobae.com

- 1 De un avión a un barco, en medio de la selva se transformó en el más impresionante caso de ingeniería: la Impresora 3D de Marte (B)
- 2 La investigación sobre la fuga del mundo de los datos de los datos y la Argentina que pone en aprietos al estado (Eugenio Cordero)
- 3 Sergio Marchionne ha estado y tiene previsto en los próximos días visitar el centro de la ciudad de Buenos Aires (B)
- 4 Las imágenes más de la casa en un barrio de la ciudad de Buenos Aires (Diego Maradona)
- 5 Un estudio que indica que el futuro de la tecnología de la información en el continente de China: "The coming of age of the Chinese" (B)

2019-07-12 10:00:00

Toyota deberá revisar 1,9 millones de autos Prius por problemas de software

Un error en el software que controla el funcionamiento del motor del auto podría hacer que se apague o funcione, pero a baja velocidad.

Compartir en Facebook

Compartir en Twitter



Toyota anunció ayer un llamado a revisión de los 1,9 millones de autos Prius de última generación vendidos en todo el mundo debido a un problema de software que provoca la caída de su sistema híbrido.

Alrededor de la mitad de los Prius a ser revisados están en Japan y 753.000 en América del Norte, dijo una portavoz de Toyota. No se han reportado accidentes relacionados al defecto, agregó.

Toyota dijo que el problema se hallaba en el software usado para controlar el conector de información en un módulo que forma parte del sistema híbrido.

"La configuración del software podría causar una mayor tensión térmica en ciertos transistores dentro del conector de información, y estos transistores podrían deformarse o dañarse como consecuencia", dijo Toyota.

"Esto resultará en el encendido de varias luces de advertencia y probablemente cause que el vehículo entre en su modalidad protegida", sostuvo la automotriz.

Agregó que en esta modalidad el coche puede conducir pero a menor potencia y velocidad de manejo.

En pocos casos, el sistema híbrido podría apagarse, lo que causaría que el vehículo se detenga, probablemente mientras está siendo conducido, dijo Toyota.

Esta reciente medida sería el tercer llamado a revisión para el Prius actual, introducido hace cinco años, después de un llamado en junio del año pasado debido a problemas ligados al acumulador de freno. El Prius es uno de los modelos más vendidos de Toyota y se ha convertido en un símbolo de la tecnología híbrida de baja emisión.

Agencia Reuters

Un error de software deja para desguace 300 coches de Subaru

Un fallo en el software de los robots soldadores de Subaru provocan la retirada de 293 coches

Por Mario Contreras 25 de mayo de 2019



Los coches se han convertido en ordenadores con ruedas y un motor. Se han convertido en una pieza central de nuestros coches que ahora, gracias a una más que dudosa seguridad, se pueden **hackear** a **distancia**. Si la programación de los coches es un problema, hay uno más grave con los robots que lo fabrican y Subaru tendrán que retirar unas 300 unidades del Ascent 2019 por un error de software.

Y sigue habiendo errores...

Y cuando
nos va bien
es por...

1. Involucramiento del usuario 15.9 %
2. Apoyo de la Gerencia 13.0 %
3. Enunciado claro de los requerimientos 9.6 %
4. Planeamiento adecuado 8.2 %
5. Expectativas realistas 7.7 %
6. Hitos intermedios 7.7 %
7. Personas involucradas competentes 7.2 %

Y cuando nos va mal es por...

1. Requerimientos incompletos 13.1 %
2. Falta de involucramiento del usuario 12.4 %
3. Falta de recursos 10.6 %
4. Expectativas poco realistas 9.3 %
5. Falta de apoyo de la Gerencia 8.7 %
6. Requerimientos cambiantes 8.1 %

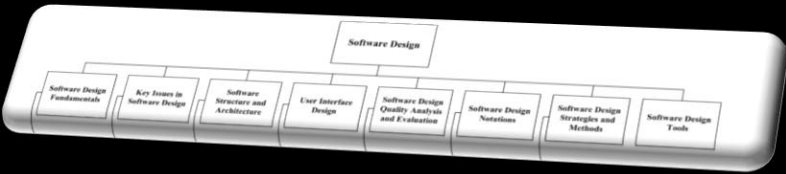
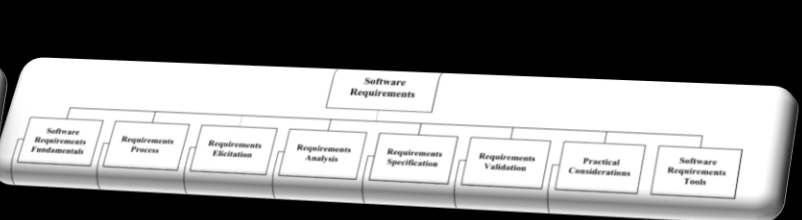
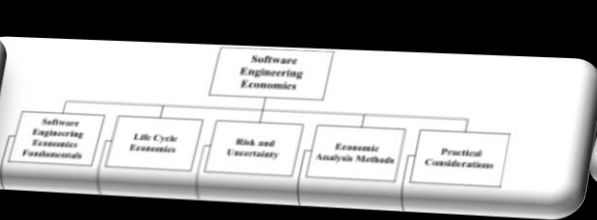


Conclusión....

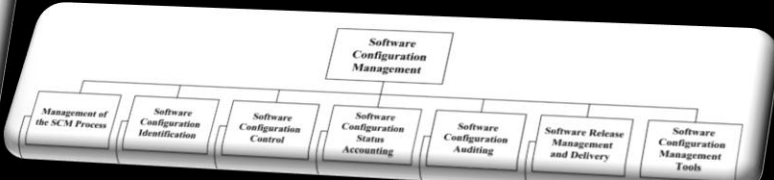
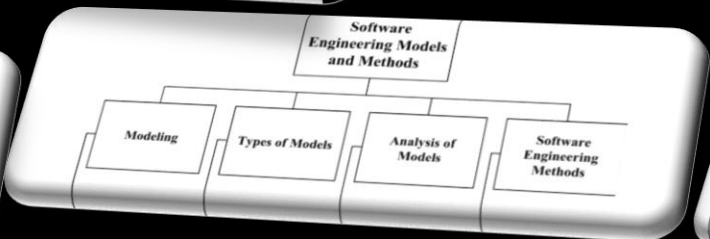
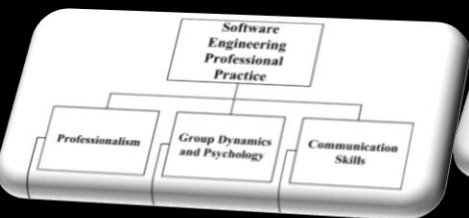
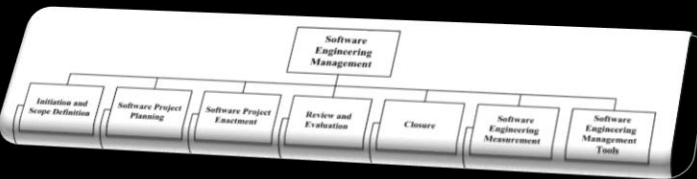
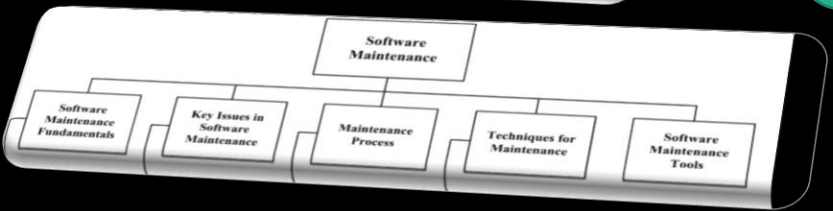
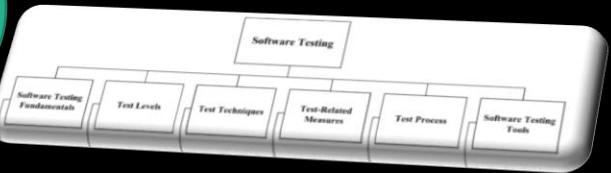
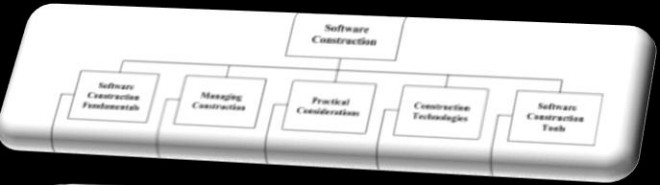
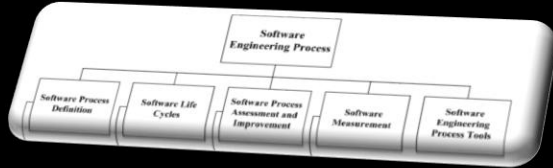
Saber programar NO es
suficiente!!!!

Ingeniería de Software

- Parmas [1987] definió a la ingeniería en software como “multi-person construction of multi-version software”



Cuerpo de
Conocimientos
de la Ingeniería
de Sw (SEBOK)





- Cuerpo de Conocimiento de la Ingeniería de Software
- Versión 3.0 del 2014 de la IEEE
- Está conformado por 15 áreas de conocimiento

Software Requirements
Software Design
Software Construction
Software Testing
Software Maintenance
Software Configuration Management
Software Engineering Management
Software Engineering Process
Software Engineering Models and Methods
Software Quality
Software Engineering Professional Practice
Software Engineering Economics
Computing Foundations
Mathematical Foundations
Engineering Foundations

Ingeniería de Software: la materia en contexto



Disciplinas Técnicas

- Requerimientos
- Análisis y Diseño
- Construcción
- Prueba
- Despliegue



Disciplinas de Gestión

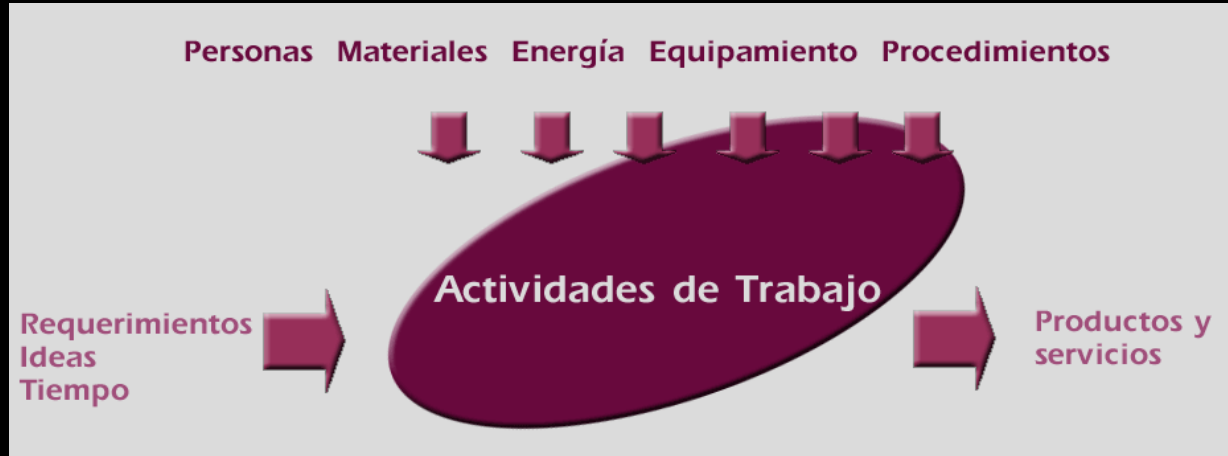
- Planificación de Proyecto
- Monitoreo y Control de Proyectos



Disciplinas de Soporte

- Gestión de Configuración de Software
- Aseguramiento de Calidad
- Métricas

El proceso de Software

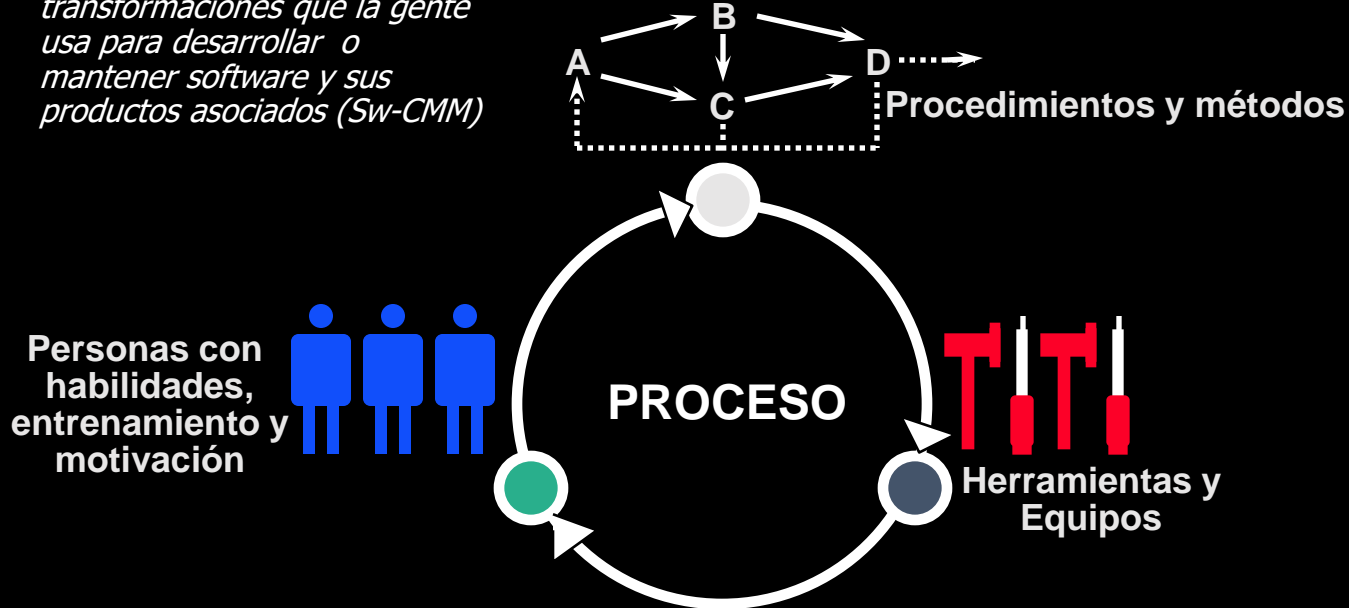


- Conjunto estructurado de actividades para desarrollar un sistema de software
- Estas actividades varían dependiendo de la organización y el tipo de sistema que debe desarrollarse.
- Debe ser explícitamente modelado si va a ser administrado.

Definición de un Proceso de Software

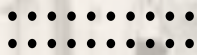
Proceso: La secuencia de pasos ejecutados para un propósito dado (IEEE)

Proceso de Software: Un conjunto de actividades, métodos, prácticas, y transformaciones que la gente usa para desarrollar o mantener software y sus productos asociados (Sw-CMM)



Definido vs. Empírico

En la Universidad de California, en Irvine, simplemente sembraron pasto y esperaron 1 año, luego de eso se fijaron donde la gente había hecho “caminito”, entonces ahí construyeron las sendas peatonales



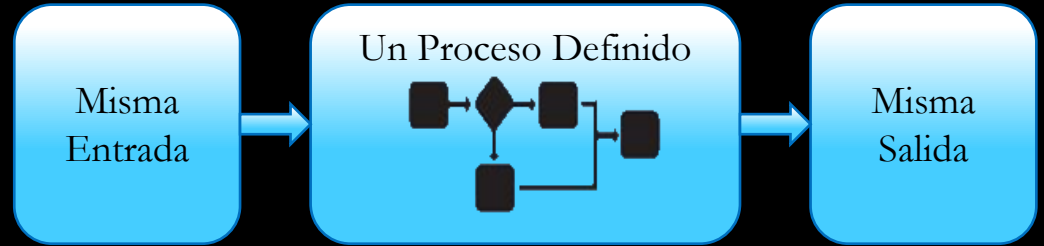
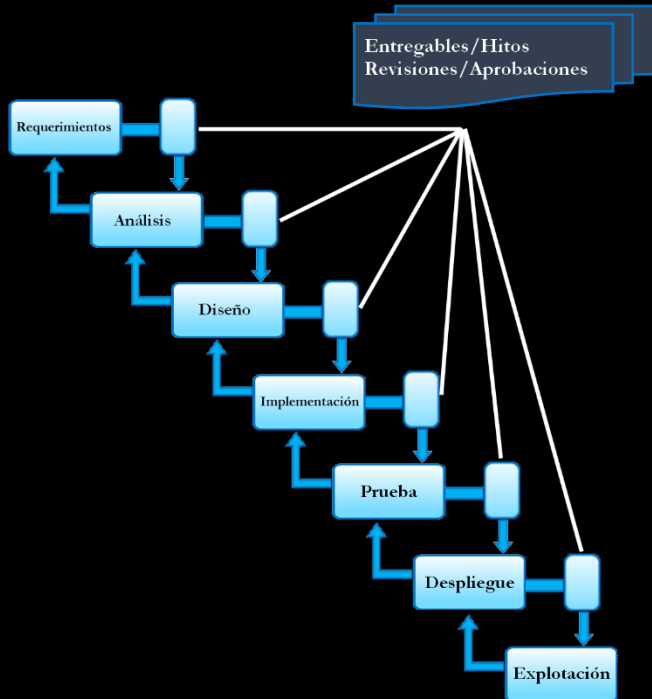
Definido (inspirados en las líneas de producción)

- Asume que podemos repetir el mismo proceso una y otra vez, indefinidamente, y obtener los mismos resultados.
- La administración y control provienen de la predictibilidad del proceso definido.

An Assembly Line
of the



Procesos Definidos



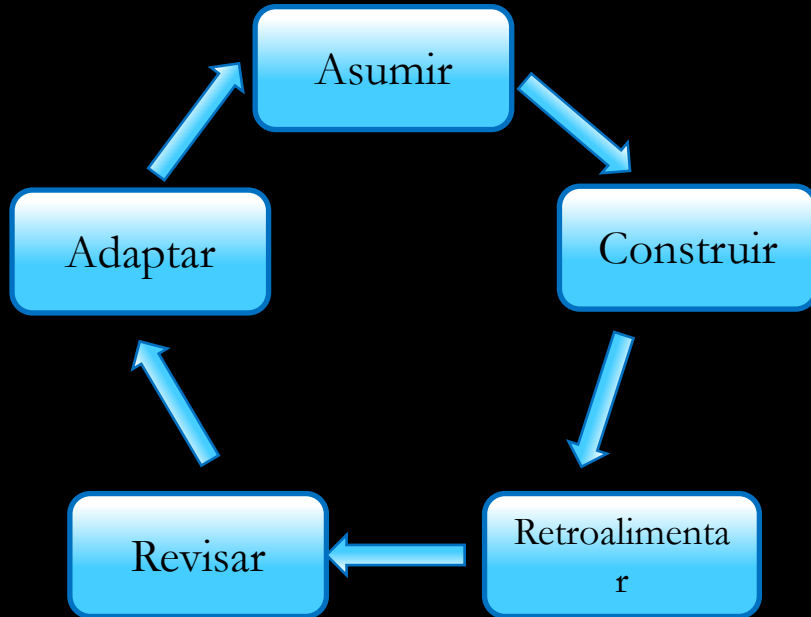
Procesos Empíricos



Empírico

- Asume procesos complicados con variables cambiantes. Cuando se repite el proceso, se pueden llegar a obtener resultados diferentes.
- La administración y control es a través de inspecciones frecuentes y adaptaciones
- Son procesos que trabajan bien con procesos creativos y complejos. (a que suena??)

Patrón de conocimiento en procesos empíricos



Ciclos de vida

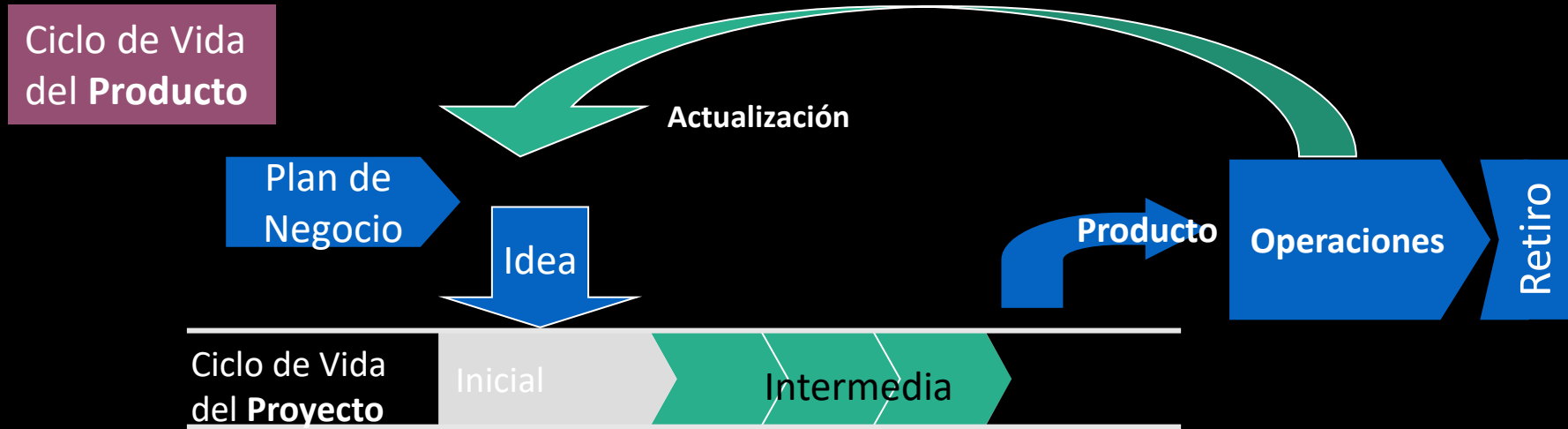
La serie de pasos a través de los cuales el producto o proyecto progresa.

Los productos tienen su ciclo de vida.

Los proyectos también.



Relación: Ciclo de Vida del Proyecto y del Producto



Ciclos de Vida de proyectos de software

- Un ciclo de vida de un proyecto software es un representación de un proceso. Grafica una descripción del proceso desde una perspectiva particular
- Los modelos especifican
 - Las fases de proceso.
 - Ejemplo: requerimientos, especificación, diseño...
 - El orden en el cual se llevan a cabo

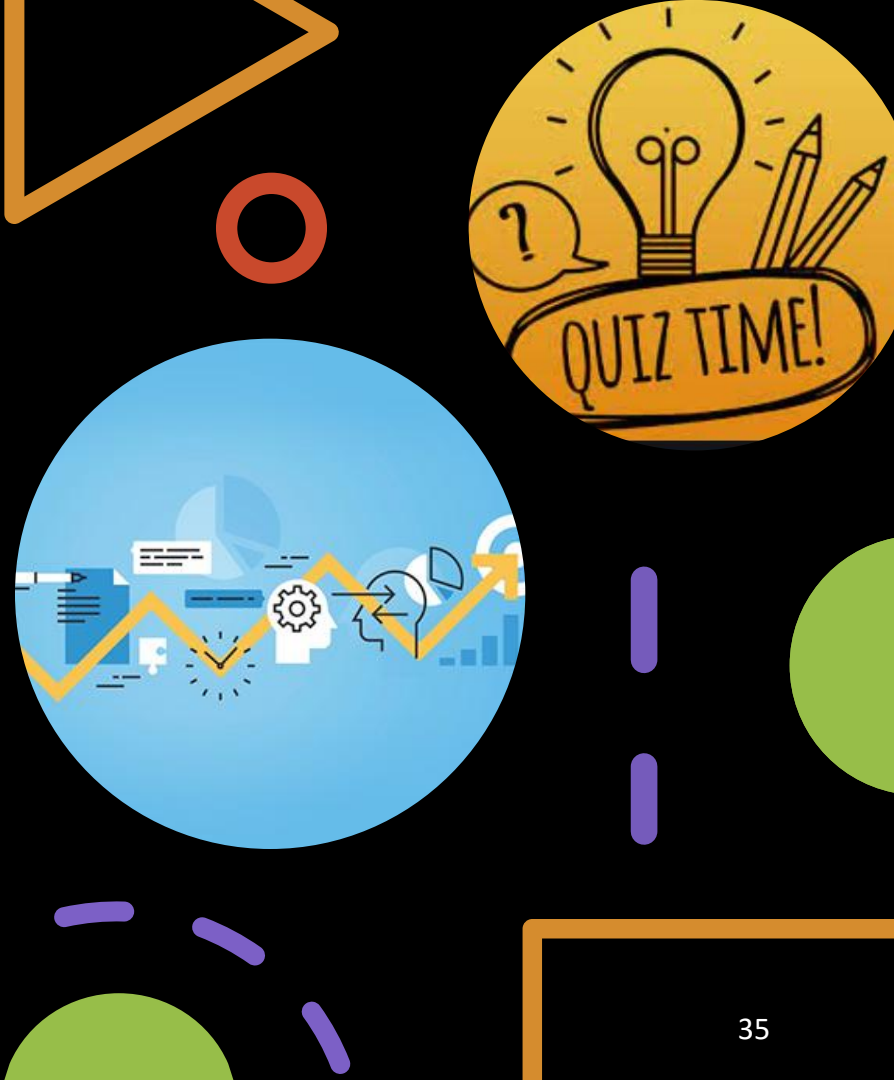
Clasificación de los ciclos de vida

- Hay tres tipos básicos de Ciclos de Vida
 - Secuencial
 - Iterativo/Incremental
 - Recursivo



¿Qué ciclos de vida conocen?

¿Qué Relación hay entre procesos de Desarrollo y ciclos de vida?



De ciclos de vida hay más?

Si! Capítulo 7 de Desarrollos de proyectos informáticos (Rapid Development) de Mcconell

Referencias Bibliográficas Principales

■ Libros:

- Sommerville, Ian - INGENIERÍA DE SOFTWARE - Novena Edición (Editorial Addison-Wesley Año 2011). -
- Brooks, Frederick The mythical man-month (anniversary ed.), 1995 Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA ©1995
- Steve Mc Connell - Rapid Development Redmond, Wa.: Microsoft Press,
- Cohn, Mike – Agile Estimation and Planning – Editorial Prentice Hall 2006
- Cohn, Mike - User Stories Applied – Editorial Addison Wesley 2004.

■ Papers:

- Orphans Preferred (<http://www.stevemcconnell.com/psd/07-orphanspreferred.htm>)
- No Silver Bullet (<http://www.virtualschool.edu/mon/SoftwareEngineering/BrooksNoSilverBullet.html>)
- Software's Ten Essentials (<http://www.stevemcconnell.com/ieeesoftware/10Essentials.pdf>)
- <http://www.scrumguides.org/download.html>
- Dean Leffingwell and Pete Behrens – A user story primer (2009)
- Manifiesto Ágil <http://agilemanifesto.org/iso/es/>

